Summary of Machine Learning Activities for Event Reconstruction

T. Suehara,

M. Kuhara, S. Tsumura, T. Onoe
(students in Kyushu)

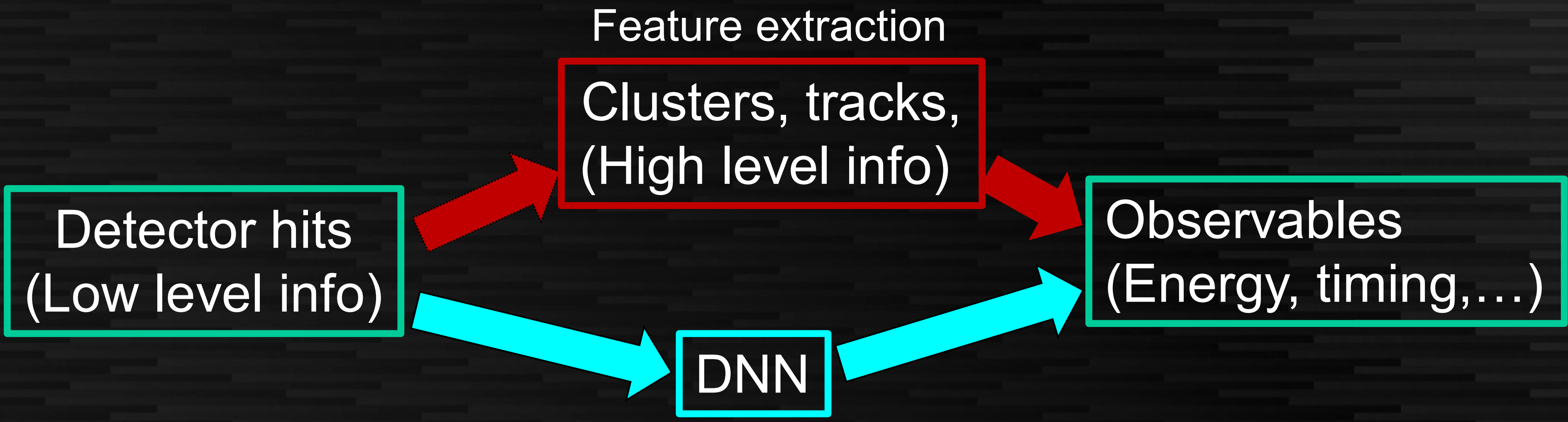
Collaboration with
M. Iwasaki (Osaka CU/Osaka U),
H. Nagahara, Y. Nakashima (IDS, Osaka U)

Topics

- Work in FY2020
  - Vertex finder with DNN (K. Goto)
    using a customized LSTM + attention
    paper under ILD collaboration review
    → submit to journal (NIM?) very soon
- Works in FY2021 and prospects in FY2022
  - Selection of timing hits in calorimeter
  - Application of HGCAL DNN reconstruction
    to ILD calorimeter
  - Flavor tagging with Graph Neural Network

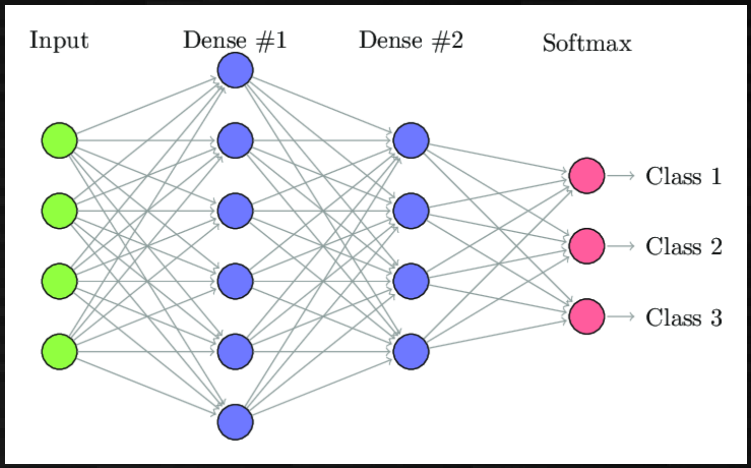
Reconstruction in Deep Learning

- With DNN (compared with good-old ML), we accept much more inputs ("big data")
- This leads "loss-less" reconstruction
  cf. usual method for "cut" or "calculate" features



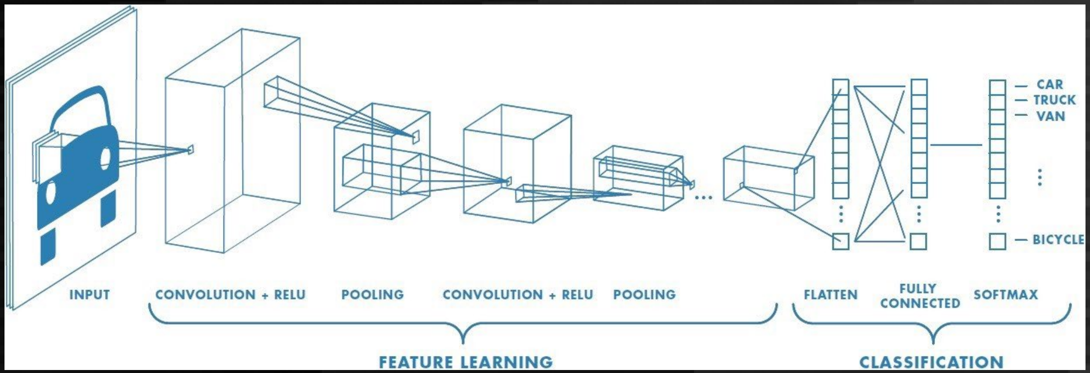
Deep Neural Networks

Fully-connected network

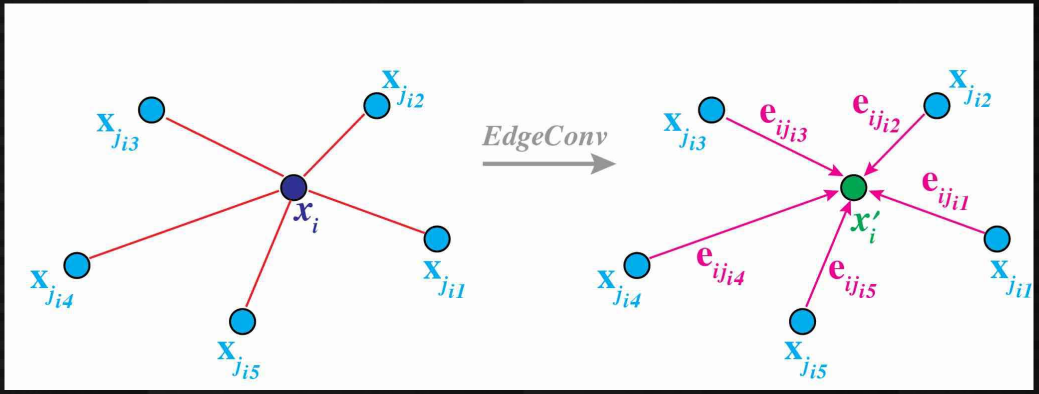


Base of all network
All nodes are equally connected
No specification on “distance”

Convolutional network (CNN)



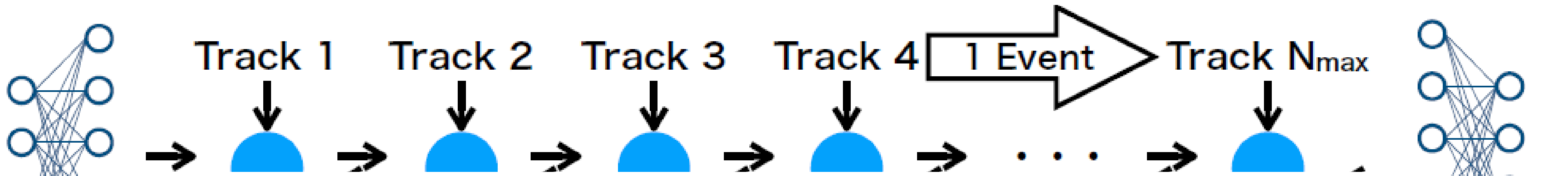
Filters connecting only to “neighbor” pixels
Suitable for image processing
Not easy for collections of (precise-but-sparse) hits



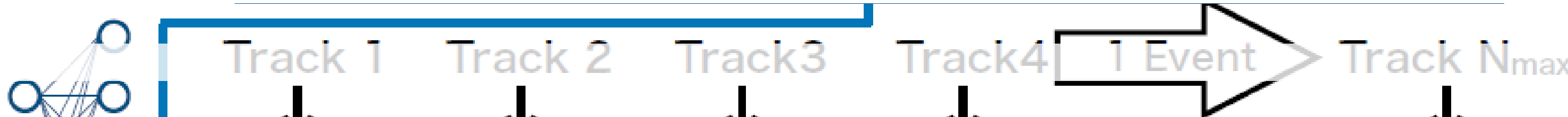
Graph neural network (GNN)

Using “connections” (edges) and/or distance between nodes
Flexible enough to use to reconstruction like PFA/flavor tag

No concrete algorithms, network design is critical



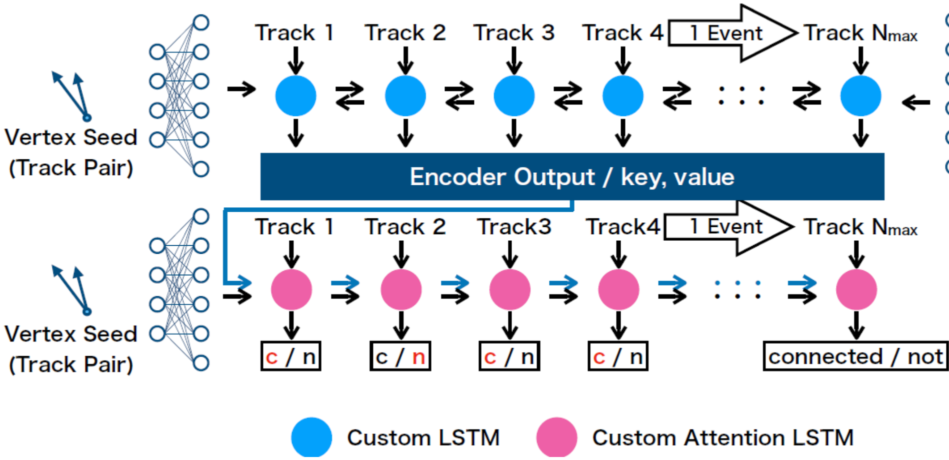
Development of a Vertex Finding Algorithm using Recurrent Neural Network



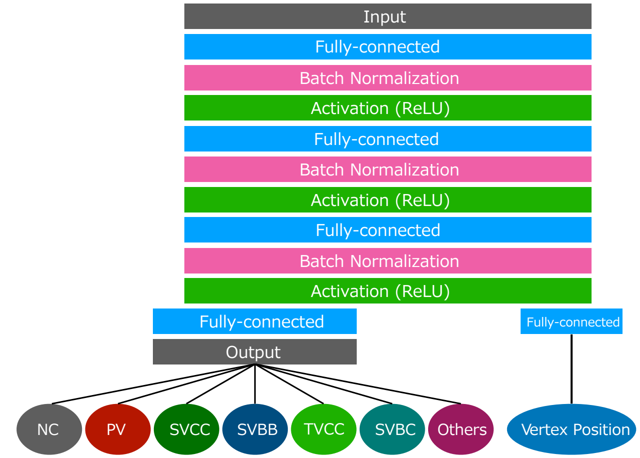
Kiichi Goto, Taikan Suehara, Tamaki Yoshioka (Kyushu U)
Masakazu Kurata (Kyushu → Tokyo → KEK)
Hajime Nagahara, Yuta Nakashima, Noriko Takemura (IDS, Osaka-U)
Masako Iwasaki (Osaka CU / Osaka U)

ILD Reviewers: Daniel Jeans (KEK), Mareike Meyer (DESY)

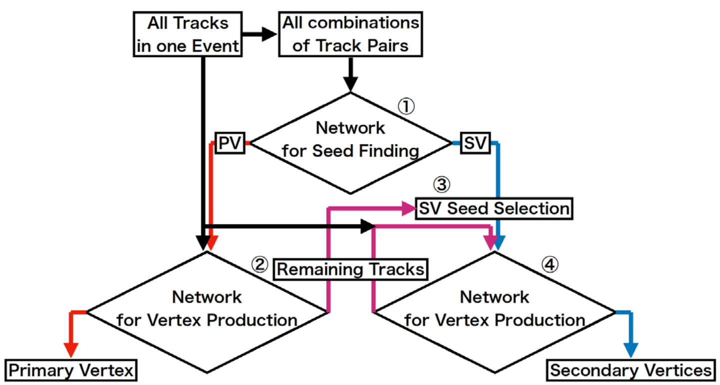
https://agenda.linearcollider.org/event/9635/contributions/50285/attachments/37947/59530/220302-dlvertex-ild-suehara.pdf



Track-assignment network



Track-pairing network



Performance similar to old one

- Better efficiency
- More contamination
  - Need V0 rejection or so
- However, flavor tagging performance not good as LCFIPlus…

Performance of DL-based vertex finder

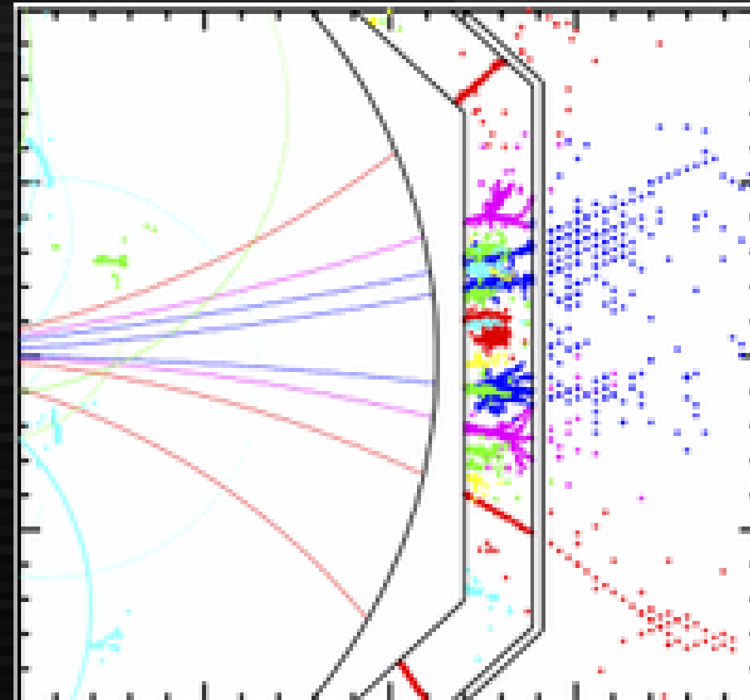
Criteria / True label	Primary	Bottom	Charm	Others
All tracks	307 657	187 283	180 143	42 888
In secondary vertex	2.2%	63.3%	68.4%	9.5%
– of same decay chain		62.3%	67.2%	
– of same parent		38.1%	36.2%	6.4%

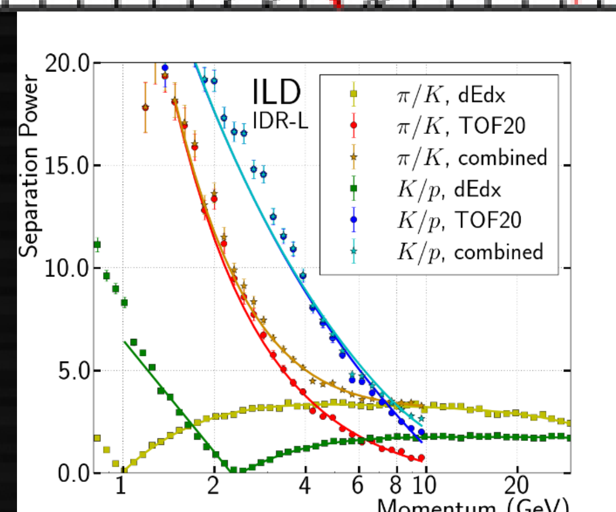
Performance of LCFIPlus vertex finder

Criteria / True label	Primary	Bottom	Charm	Others
All tracks	307 657	187 283	180 143	42 888
In secondary vertex	0.2%	57.9%	60.3%	0.5%
– of same decay chain		57.5%	59.9%	
– of same parent		34.0%	37.2%	0.3%

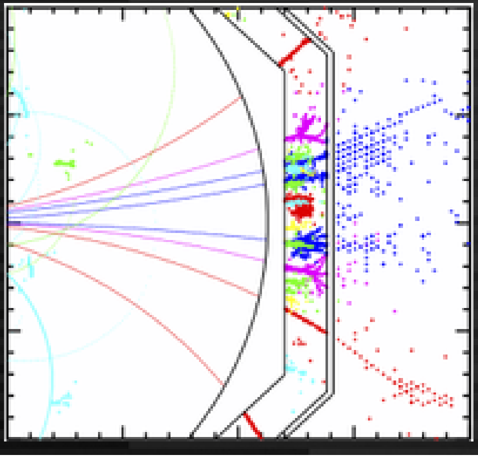
Timing reconstruction

- Time-of-Flight (ToF) is a powerful tool for hadron ID (π/K/p separation)
  - < 20 psec required
- At calorimeters, hits can be averaged to improve timing resolution of the sensors
- Hadrons @ ECAL
  - Track-like: easy to average
  - Track + 2ndaries
    - Have to identify path inside CAL
  - Showering
    - Separate usable/unusable hits

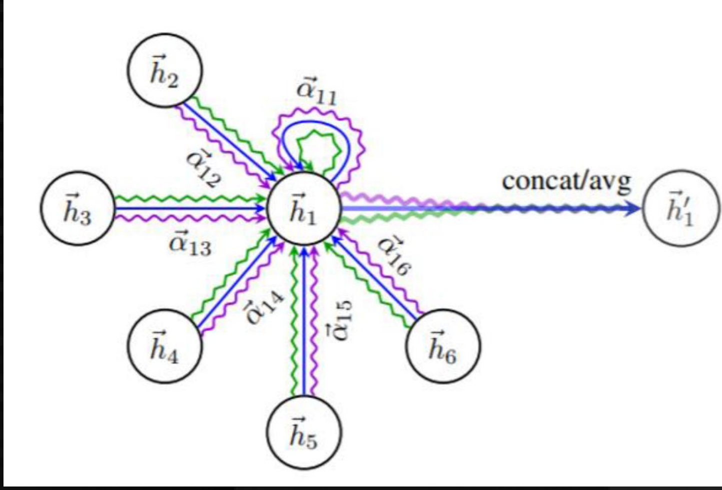




Timing by Graph Attention Network



- Input variables
  - Position, timing (smeared), energy deposit of each hit
- Possible output
  - Selection of "prompt" hits
  - Ordering of hits (parent hit)
  - Averaged time at surface
- Structure
  - Graph attention network
    - Optimize "connection strength" between nodes (hits)
  - Supervising connection (attention weights) or nodes after processing



GAT

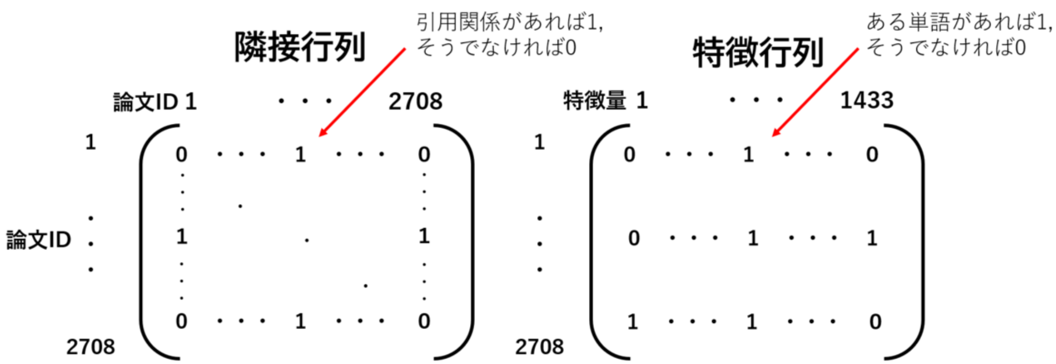
Graph Attention Network (GAT) (arXiv:1710.10903)

- Learn "connection matrix" (attention) among nodes and update features with the attention matrix

eg.) Categorize published papers (nodes) by

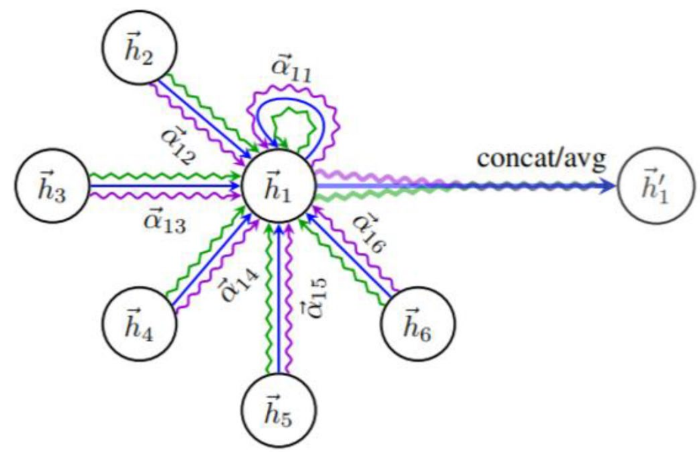
Citing relation as adjacent matrix and

Words used in the paper as features of the nodes



Adjacent matrix
Define connecting nodes

Features in each node



$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$

input data

Adjacent matrix

- [300,500,500]

  ([event,hit,hit])
- Zero-padding for <500 hits
- Currently all 1 with existing hits

  → Restriction on distance?

feature matrix

- [300,500,7]

  ([event,hit,feature])
- 7 features
  - Index
  - Layer
  - mc time (can be smeared)
  - Energy
  - Positionx,y,z

label (answer for learning)

- [300,500]

  ([event,hit])
- 0/1 binary
- Set 1 if difference between hit time (without smearing) and expected time from track history (parent-daughter relation by MC info) is within 20 psec,
  0 otherwise
  Cutting slow hits not usable for the time reconstruction

100 events for training, 100 for validation, 100 for test
(Need to implement mini-batch training to accommodate more events)

STATUS

Network

FC only: Fully-connected(7, 8) → Fully-connected (8,2) → Softmax

Concat: FC concatenated with GAT model

Loss: cross entropy, Accuracy: fraction of correct categorization

after 1000 epochs		loss_train	acc_train	loss_val	acc_val	loss_test	acc_test
mc time not smeared	concat	0.1878	0.9738	1.0282	0.8976	1.0467	0.9216
	FC only	0.2526	0.9809	0.3920	0.9644	0.4766	0.9448
mc time 100 ps smeared	concat	0.2853	0.9592	0.8141	0.9079	0.8621	0.9349
	FC only	0.3523	0.9559	0.4949	0.9447	0.4933	0.9376
mc time = 0	concat	0.2027	0.9358	0.7610	0.8794	0.7314	0.8897
	FC only	0.3401	0.9388	0.3898	0.9286	0.4580	0.9144
mc time, Index = 0	concat	0.1770	0.9409	0.2907	0.9240	0.3638	0.9222
	FC only	0.3353	0.9429	0.3619	0.9368	0.3943	0.9178

features

- Index
- Layer
- mc time
- Energy
- Positionx,y,z

mc time smeared with Gaussian

No significant difference currently. Need more investigation.

PROSPECTS

Short-term

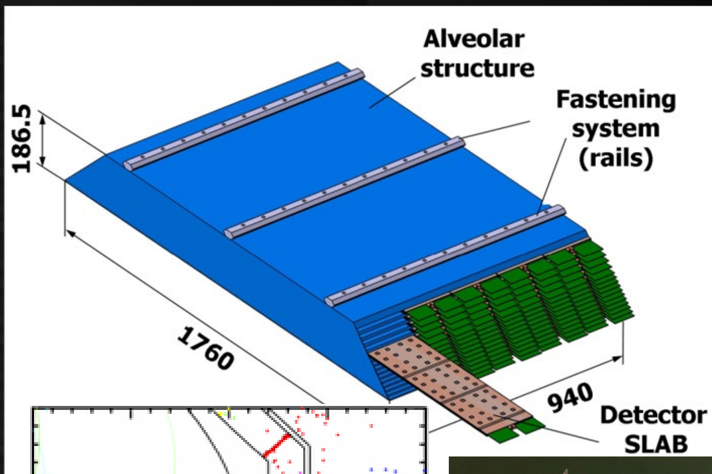
- Implement mini-batch training
- Current accuracy uses all 500 hits → modify to use only existing hits
- Shuffing hit ordering
- Normalization of input values
- Optimize attention eg. by training MC-truth relation

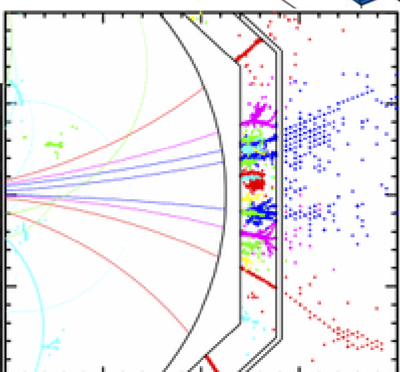
Long-term

- Establish selection of prompt hits with this method
- Apply also to the "tracking" of showers (parent-daughter relation)
- Apply also to the timing calculation
  (or calculate timing without DNN with results above)
- Apply timing resolution dependent of signal strength
- Goal: ~1/10 of single-hit timing resolution (depending on number of hits)

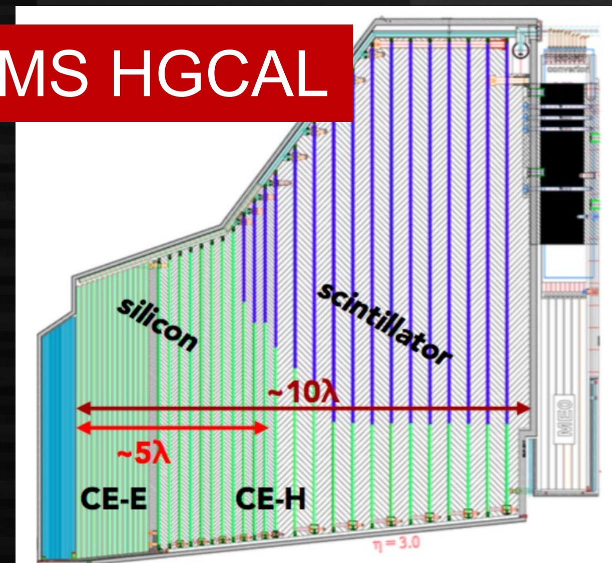
ILC SiW-ECAL / CMS HGCal

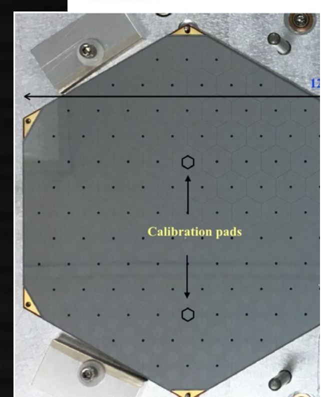
ILD SiW-ECAL

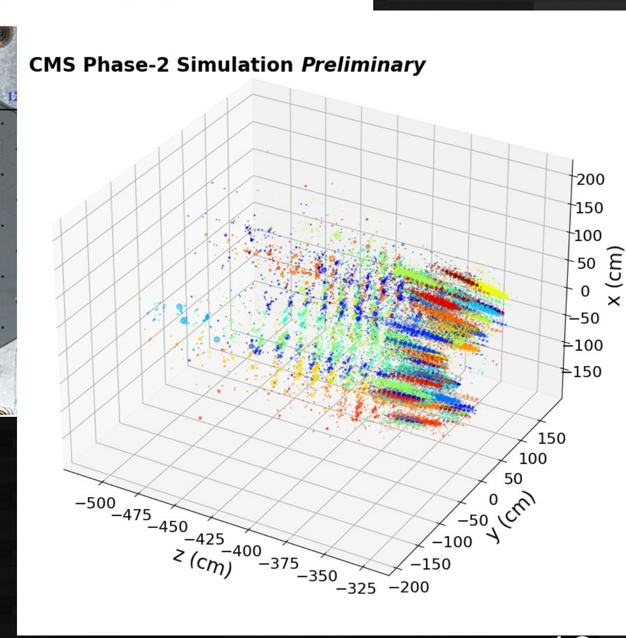




CMS HGCAL

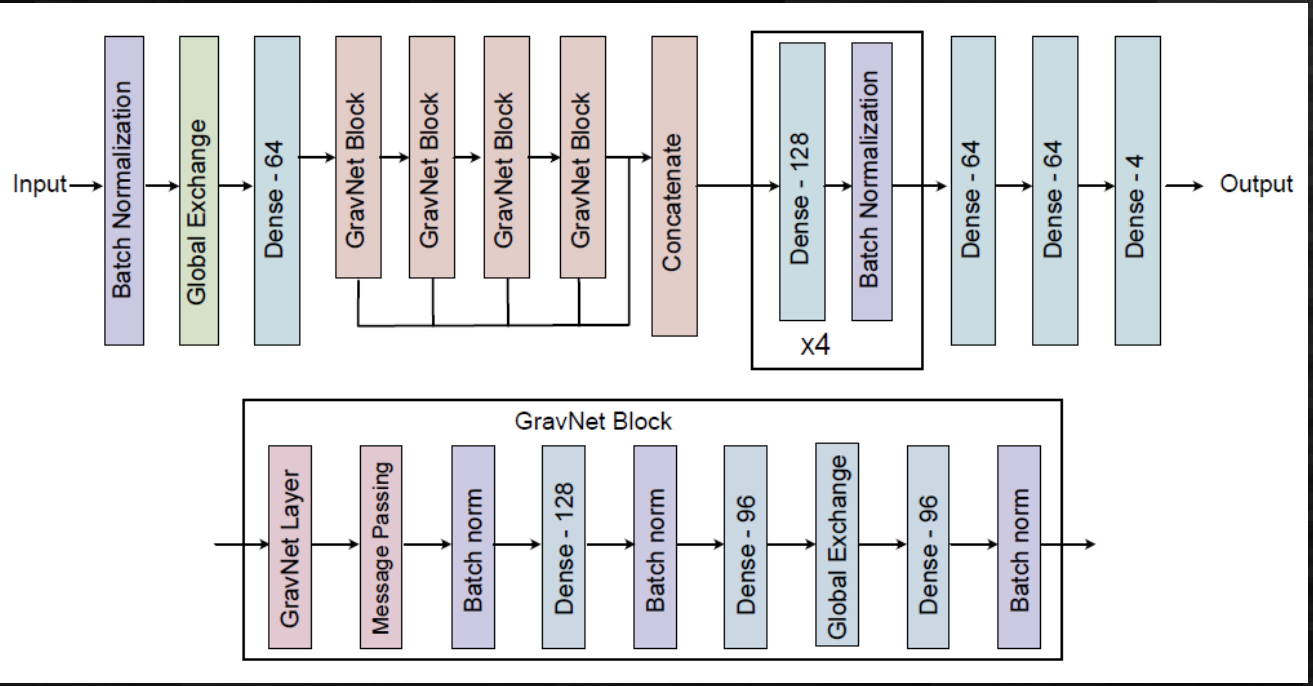






CMS HGCAL is developed based on CALICE SiW-ECAL development
→ similar structure

HGCAL reconstruction network



Around 30
hidden layers

Clustering objects (PFOs) from collection of hits

Input: Variables of each hit (position, energy, timing)

Output: 4 per hit (Position in abstract coordinate: 2,
Object condensation: 1, Energy: 1)

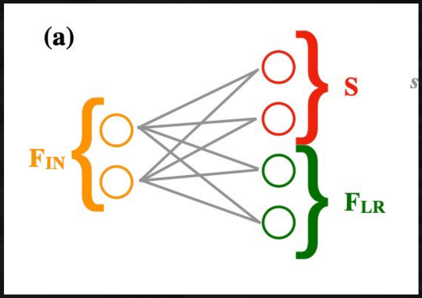
Dense: Fully-connected layer inside the hit

Batch norm: Normalization method for avoiding over-training

GravNet / Object Condensation

GravNet

1. Feature of hits are treated with single NN to abstract position S and features F_{in}



2. Distance at S coord. used to combine F of neighbors (F_{LR}', F_{LR}')
3. Combine F_{in}, F_{LR}', F_{LR}'' using FC layer (F_{out})

Object Condensation

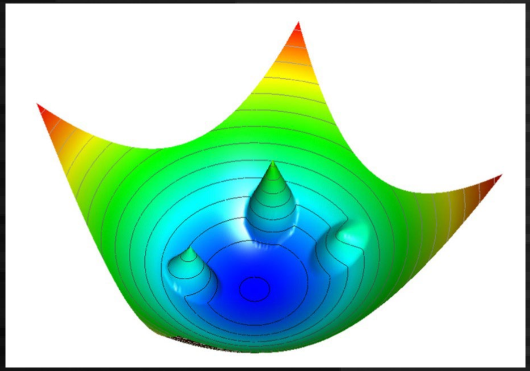
Used as a loss function

$$L = L_p + s_C(L_\beta + L_V)$$

Minimize L

β: Condensation parameter

L_v: attractive force for high β and from same MC particle, repulsive force for high β and from another MC particle

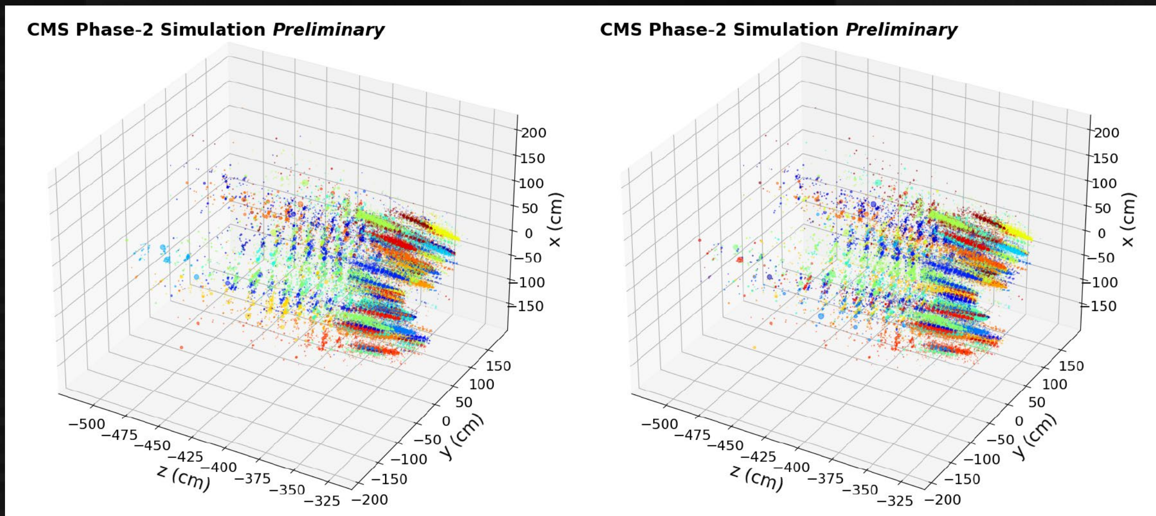


L_β: β going to 1 for cluster, 0 for noise

L_p: Used for energy regression

s_c: hyperparameter

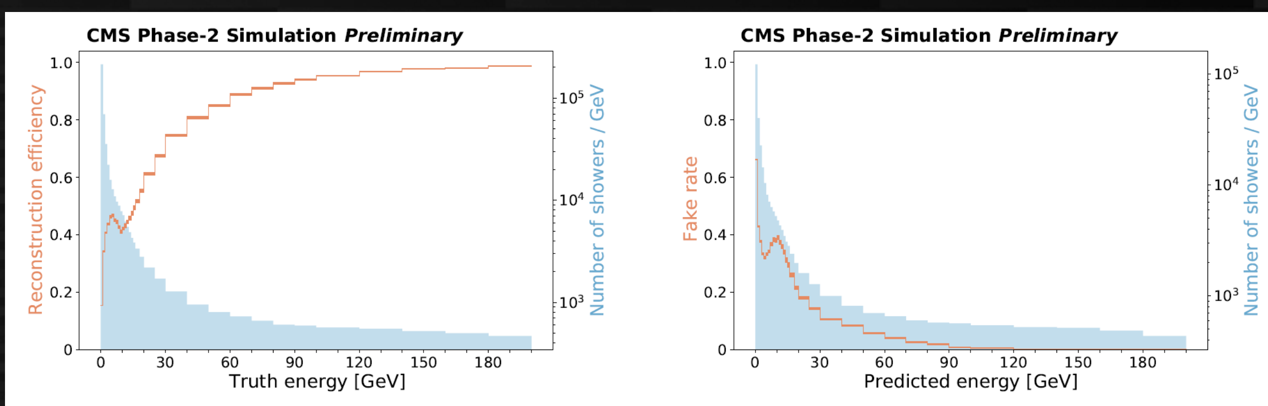
Performance at HGCAL



100 particle
overlaid randomly

left: true cluster
right: reco cluster

Good agreement



(a) The reconstruction efficiency as a function of the true cluster energy

(b) The fake rate as a function of the reconstructed cluster energy

left: efficiency
right: fake rate

good quality
for high energy

Status for ILC application

- Code itself is public
  - https://github.com/cms-pepr
- Having C++ port (probably for speed)
  - Need special environment for compiling
  - Not succeeded yet
    - Some issues on version mismatch
      - libtorch, gcc, python, cuda, …
- Input should be prepared as well

Prospects

- HGCAL reconstruction with ILD simulation
  - Prepare hits as input
  - MC truth for cluster: non-trivial (definition of cluster)
- Peformance comparison with PandoraPFA for clusters
  - Using Pandora cluster output?
  - Inference by Pandora module?
    - Possible to use Pandora track-cluster matching for full PFA
- Performance optimization
  - Hyperparameter tuning, network structure optimization
  - Advanced options like multi-step training, transfer traning etc.

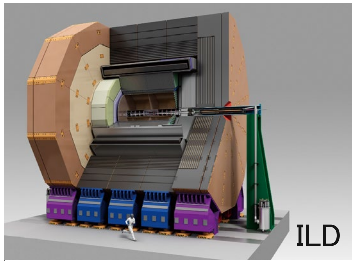
US-J proposal submitted with FNAL (CMS)

Final topic: flavor tagging

- LCFIPlus: used from 2012 (10 years old!)
  - T. Tanabe and TS
  - Now maintained by R. Yonamine
- DNN result becoming popular in flavor tagging
  - LHC experiments
- Need to modernize LCFIPlus (LCFI++?)
  - Combining vertex finding and flavor tagging
  - Low level input, no loss of information

Data

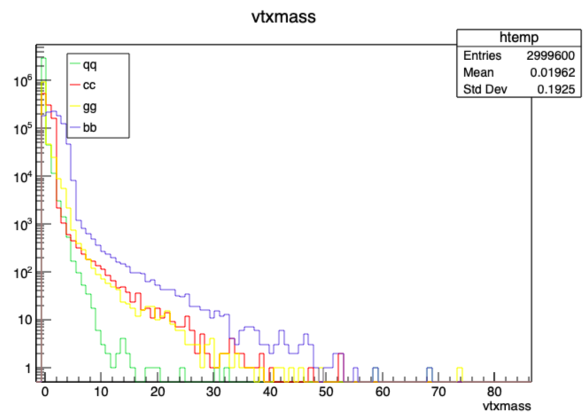
Data preparation



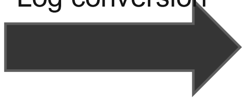
- 250GeV ILD simulation, 4M events
- 124 variables of input of LCFIPlus
  (例. # of vertex, position, mass, probablility, tracks inside vertices,
  impact parameter significance etc..)

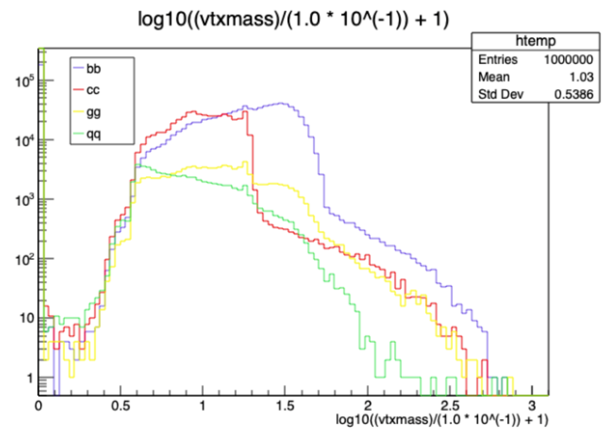
Preprosessing

NN is efficient with same-order-of-magnitude input → conversion by log etc.

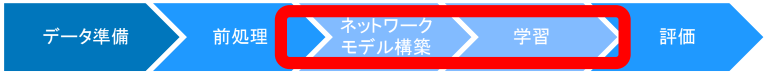


Log conversion





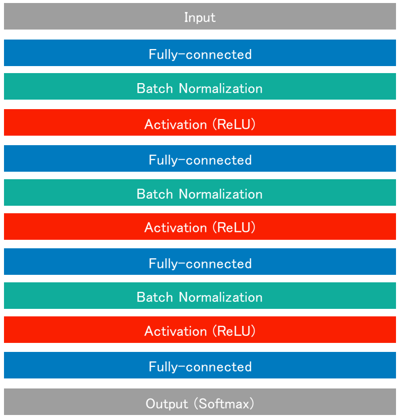
Models and learning



Network model

- simple DNN as a basis for comparison to modern network

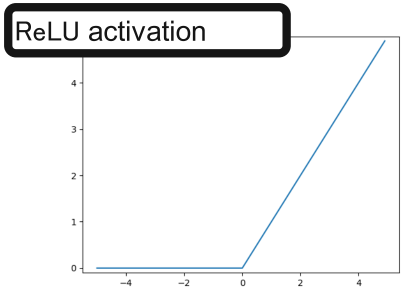
- Input : 42 parameters (used in LCFIPlus)
- 4 FC layers with batch norm and dropout
- ReLU for activation
- output : 3 categories (b, c, uds)



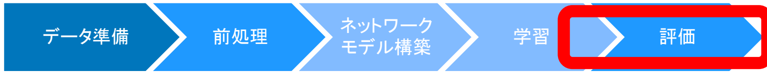
Learning

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\\text{size}}} y_i \cdot \log \hat{y}_i$$

- Loss function : Categorical Cross Entropy
- Optimizer : Adam (LR : 0.01 → reduced by 10 by 25 epochs)
- Max epochs : 100 epochs
- Batch size : 1024

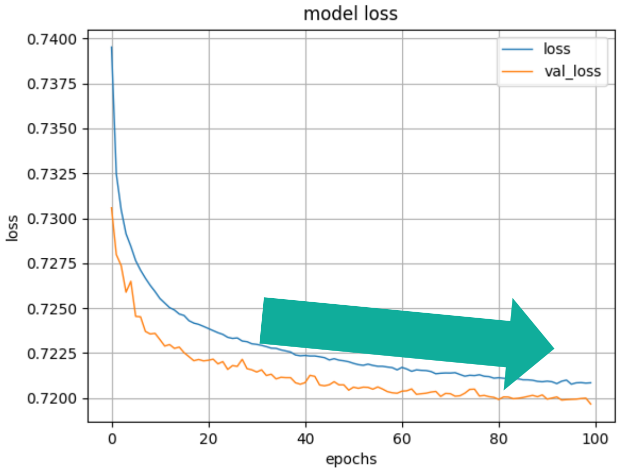


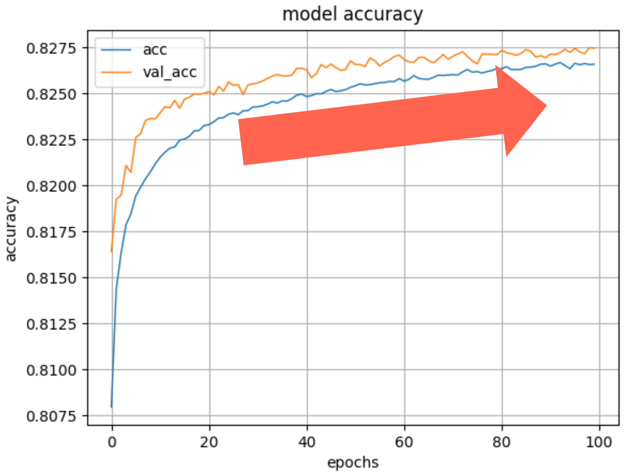
Results (loss and accuracy)

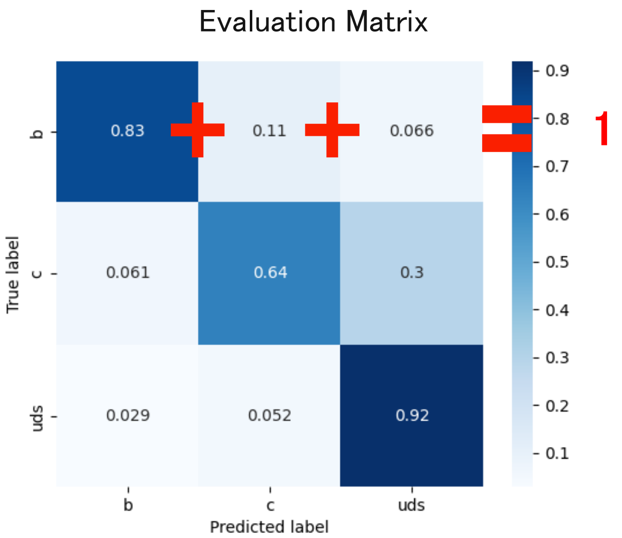


Evaluation

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$





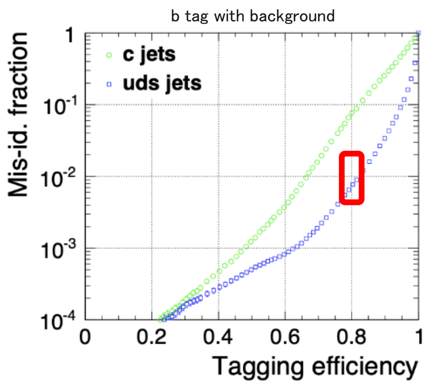


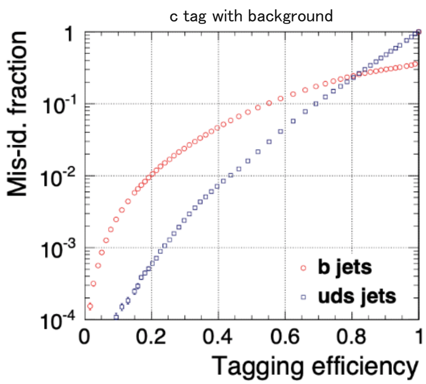
Difference of loss for training and validation samples goes smaller with more epochs → good response

Results (flavor-tag performance)

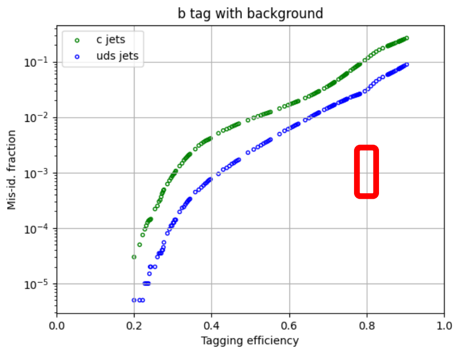
データ準備 → 前処理 → ネットワーク モデル構築 → 学習 → 評価

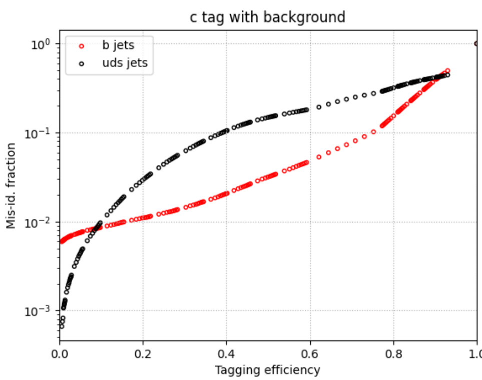
LCFIPlus





DNN





Graph neural network for flavor tagging

- Full reconstruction from low level data (tracks, neutral particles)
- Utilize geometric information (not just FC network)

Graph Neural Network (GNN)

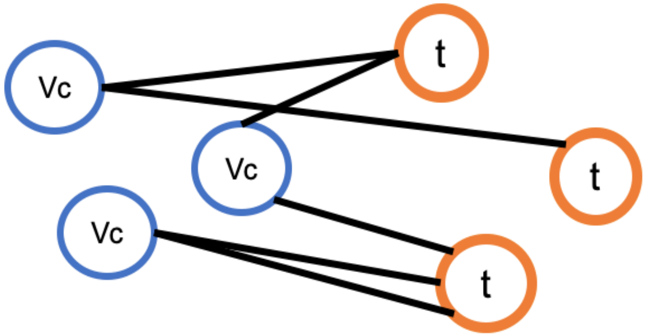
Input

Graph : 2 kinds of nodes and edges

Nodes : Track

Vertex Candidate (VC) with all track pair

Edges : Connecting VC and tracks



Output

Graph classification (b/c/uds): final goal

Node classification (true/fake vertex, b/c/primary vertex)

Combination of VC to reconstruct vertices?

Detailed graph models are under investigation

Summary

- Various works with DNN for event reconstruction is ongoing
  - Graph-based approach being tested: need to decide detailed structure (no universal strategy in GNN)
    - Graph Convolutional Network
    - Graph Attention Network
    - GravNet/Object condensation (CMS method)
    - Many many others
    - Transformer (more flexibility?)
  - Intensive studies still needed for better reconstruction
    - More (professional) manpower desired